[redacted]

UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF TEXAS
SHERMAN DIVISION

R2 SOLUTIONS LLC,

Plaintiff,

v.

DATABRICKS, INC.,

Defendant.

Civil Action No. 4:23-cv-01147-ALM

JURY TRIAL DEMANDED

[redacted]

DATABRICKS, INC.'S MOTION FOR SUMMARY JUDGMENT OF NON-INFRINGEMENT

[redacted]

TABLE OF CONTENTS

[redacted]

TABLE OF AUTHORITIES

[redacted]

TABLE OF EXHIBITS[1]

No.	Exhibit
1	U.S. Patent No. 8,190,610 (the "'610 patent")
2	Excerpts of 2024 Databricks customer-facing slide deck (Databricks_R2_00091426 – Databricks_R2_00091494)
3	Databricks webpage titled "What is Photon?" (GMacartney_00000892 – GMacartney_00000895)
4	Expert Report of William Davis Concerning Infringement of U.S. Patent No. 8,190,610 ("Davis Op.")
5	Rebuttal Expert Report of Dr. Jon Weissman ("Weissman Rebut.")
6	Excerpts of Transcript of the March 25, 2025 Deposition of William Davis ("Davis Dep. (Day 1)")
7	Excerpts of Transcript of the December 13, 2024 Deposition of Cody Davis ("C. Davis. Dep.")
8	Excerpts of Transcript of the January 16, 2025 Deposition of Joshua Rosen ("Rosen Dep.")
9	Excerpts of Transcript of the January 24, 2025 Deposition of Reynold Xin ("Xin Dep.")
10	Excerpts of Transcript of the January 15, 2025 Deposition of Vinod Nair Viswanath (Comcast) ("Viswanath Dep.")
11	Excerpts of Transcript of the January 8, 2025 Deposition of George Sirois (Abnormal Security) ("Sirois Dep.")

[1] Each of the exhibits listed in this table, Exhibits 1-11, is attached to the Declaration of Vigen Salmastlian.

[redacted]

I. INTRODUCTION

This case is ripe for summary judgment. R2 asserts claims 1, 5, 17 and 21 of U.S. Patent No. 8,190,610, which require specific input data groups and a specific set of MapReduce data processing steps. R2 alleges only literal infringement by the Databricks Data Intelligence Platform ("Databricks platform"), and thus, does not assert infringement under the doctrine of equivalents.

It is undisputed that the Databricks platform includes a set of tools that Databricks customers and users may choose to implement in whatever way they desire. The Databricks platform *does not perform* data processing out of the box. To perform data processing, a Databricks user must provide (1) its own data sets, (2) its own data processing source code that defines what data processing to perform on those data sets, and (3) its own configuration selecting how to perform that data processing. But there is no evidence in this case of any Databricks user's data sets, data processing code, or configuration of the Databricks platform. Faced with this total lack of evidence and a job to prove infringement, R2's expert Mr. Davis *created* his own data sets, *created* his own code, and *assumed many conditions* in his configuration of the Databricks platform, including turning off the default Photon data processing engine. Mr. Davis did this to force the Databricks platform into a configuration that uses the Apache Spark data processing engine to select a specific join strategy for joining two data sets called "Sort Merge Join." That is the focus of Mr. Davis's concocted theory—use of Spark's "Sort Merge Join" in the specific way Mr. Davis came up with allegedly infringes the asserted claims. But as Mr. Davis testified, not only does he have no evidence that any user does what he did to show infringement, there are other join strategies that users may choose to implement and for which he has no infringement opinion. There is therefore no evidence of any direct infringer under Mr. Davis's concocted theory.

Moreover, R2 cannot prove infringement of the new generation of the Databricks platform called "serverless compute" because it undisputedly does not use the allegedly infringing Spark

[redacted]

"Sort Merge Join." In serverless compute, the Photon data processing engine is on and *cannot be turned off*. It replaces the accused Spark "Sort Merge Join" with Photon functionality for which Mr. Davis has no infringement opinion.

R2 also cannot show that the earlier generation of the Databricks platform called "classic compute" infringes system claims 17 and 21 because Databricks does not make, sell, or use the claimed system. Indeed, claim 17 as construed by the Court, requires a "computer system" with "at least one processor and memory that are operable to perform" the MapReduce data processing steps of the claim. But Databricks users must purchase "processor and memory" resources from a third-party cloud provider (such as Amazon, Google, or Microsoft), provide their own data sets, create their own data processing code, and configure the Databricks platform for the data processing they want to perform. A Databricks user must do all of these steps to create the claimed "computer system" and cause a cloud provider's "processor and memory to be operable to perform" any data processing at all. Thus, Databricks does not make or sell the claimed "computer system." R2 also has no evidence that any user has ever "used" the claimed "computer system" to perform the claimed steps in the way Mr. Davis alleges.

Finally, even under R2's concocted infringement theory, there is no dispute of fact that R2 cannot show the accused Databricks platform meets three of the Court's claim constructions.

II. STATEMENT OF THE ISSUES

1. Whether R2 has any evidence sufficient to raise a genuine dispute of material fact that Databricks serverless compute infringes any asserted claim given that it implements Photon functionality that replaces the Spark "Sort Merge Join" strategy that R2 relies on for infringement.

2. Whether R2 has any evidence sufficient to raise a genuine dispute of material fact regarding direct infringement given that R2 does not identify any Databricks customer or user that has implemented the configuration Mr. Davis created and relied on for infringement.

[redacted]

3. Whether R2 has any evidence sufficient to raise a genuine dispute of material fact that Databricks classic compute directly infringes any asserted system claim given Databricks does not make or sell any "computer system" with the required "processor and memory," and R2 has no evidence of any "use" implementing the allegedly infringing configuration Mr. Davis created.

4. Whether R2 has any evidence sufficient to raise a genuine dispute of material fact that the Databricks platform infringes any asserted claim given that R2's expert admits the accused "data group" does not include the "mechanism for identifying data from that group" as required by each asserted claim under the Court's construction.

5. Whether R2 has any evidence sufficient to raise a genuine dispute of material fact that the Databricks platform infringes any asserted claim given that R2's expert admits that the accused "mapping function" is not selected for a partition based on the accused "data group" that a partition originates from, as required by each asserted claim under the Court's construction.

6. Whether R2 has any evidence sufficient to raise a genuine dispute of material fact that the Databricks platform infringes any asserted claim given that R2's expert admits that the accused "reducing" does not process intermediate data for each accused data group "in a manner that is defined to correspond to the data group from which the intermediate data originated," as required by each asserted claim under the Court's construction.

III. STATEMENT OF UNDISPUTED MATERIAL FACTS

A. The asserted patent

7. The '610 patent is entitled "MapReduce for Distributed Database Processing," and relates generally to an "enhanced MapReduce programming methodology." (Ex. 1, '610 patent, Title, 1:42-43.) MapReduce is a prior art "programming methodology to perform parallel computations over distributed (typically, very large) data sets" using two functions: map and reduce. (*Id.* at 1:6-8, 1:17-20.) A "'map' function maps key-value pairs to new (intermediate)

[redacted]

key-value pairs" and a "'reduce' function represents all mapped (intermediate) key-value pairs sharing the same key to a single key-value pair or a list of values." (*Id.* at 1:17-20.) The Court construed "reducing" consistent with this lexicography in the patent. (Dkt. 71 at 21.) The patent describes a purportedly "improved MapReduce architecture" that provides "a mechanism to associate (group) identifiers with data sets, map functions and iterators (usable within reduce functions to access intermediate data) and also to produce output data sets with (group) identifiers." (Ex. 1, '610 patent, 3:48-50, 3:58-62.) This allows using different "map tasks 402 and 404 [that] are configured to operate on separate data groups…characterized by [their] own schema."[2] (*Id.* at 3:65-4:1.) "[F]or a particular group identified with a group_id, the map function processes each input key/value pair and produces one or more intermediate key/value pairs." (*Id.* at 4:48-50.) And "the reduce [] merges all intermediate data over all groups." (*Id.* at 4:56-57.)

8. R2 asserts claims 1, 5, 17, and 21. (Ex. 4, Davis Op., ¶¶ 36-37.) All asserted claims require the data processing steps "partitioning," "mapping," and "reducing" that are performed on a "plurality of data groups" that have "different schema." (Ex. 1, '610 patent, claims 1, 17.) The Court construed "data group" as "a group of data and a mechanism for identifying data from that group" and "schema" as "plain and ordinary meaning." (Dkt. 71 at 6, 34.) Additionally, the "map" step requires, in part, "providing each data partition to a selected one of a plurality of mapping functions…to form corresponding intermediate data for that data group." (Ex. 1, '610 patent, claims 1, 17.) This "providing…" limitation was construed as "providing each data partition to one of a plurality of different mapping functions where each mapping function is selected for a partition based on the data group the partition originated from." (Dkt. 71 at 27.) The "reduce"

[2] Employee table 302 and Department table 304 of Figure 3 have different "schema" because they include different "set[s] of attributes (such as DeptID, LastName DeptName) and their properties (such as their data types: integer DeptID, string LastName, string DeptName)." (*Id.* at 3:37-41.)

[redacted]

step requires, in part, "reducing the intermediate data for the data groups to at least one output data group, including processing the intermediate data for each data group in a manner that is defined to correspond to that data group." (Ex. 1, '610 patent, claims 1, 17.) This "processing…" limitation was construed as "processing the intermediate data for each data group in a manner that is defined to correspond to the data group from which the intermediate data originated." (Dkt. 71 at 30.)

B. The accused Databricks platform

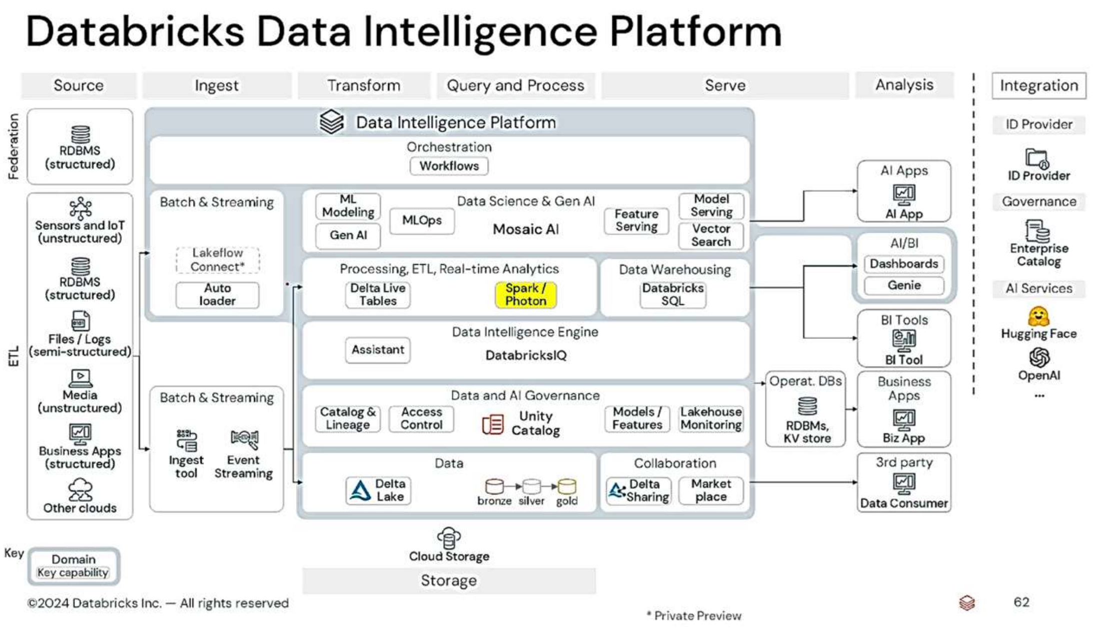
9. The Databricks platform is a set of software tools for large-scale data processing, machine learning applications, data management, and other tasks. (Ex. 5, Weissman Rebut., ¶¶ 88-90; Ex. 6, Davis Dep. (Day 1), 14:7-15.) Databricks users choose which of these tools to implement and how to implement them based on their specific needs. (*Id.*) For example, the Databricks platform offers a tool called Notebook, which provides an interface that allows Databricks users to write code to run data processing queries in the Databricks platform. (Ex. 4, Davis Op., ¶ 71; Ex. 5, Weissman Rebut., ¶¶ 97-98.) Databricks users must write code in some form to execute a data processing query because the Databricks platform does not perform any data processing out of the box. (Ex. 6, Davis Dep. (Day 1), 13:14:6 ("[The user] has to give some code."); Ex. 4, Davis Op., ¶ 71; Ex. 5, Weissman Rebut., ¶ 236.) Databricks users must also provide the data sets they would like to analyze. (Ex. 6, Davis Dep. (Day 1), 18:20-19:1.)

10. There are two generations of the Databricks platform: classic (or non-serverless) compute and serverless compute. In "the original version of the Databricks platform called 'classic compute' customers pay a cloud provider directly for their usage of the cloud provider's resources." (*Id.* at 17:22-18:2.) Databricks users must purchase these cloud resources directly from a cloud provider: Amazon's Amazon Web Services ("AWS"), Microsoft's Azure ("Azure"), or Google's Google Cloud Platforms ("GCP"). (Ex. 4, Davis Op., ¶ 70; Ex. 5, Weissman Rebut.,

[redacted]

¶ 88.) That is because "Databricks itself does not provide processors or memory resources for classic compute." (Ex. 6, Davis Dep. (Day 1), 18:4-7.) In other words, to use the Databricks platform for classic compute, a user must create an account with one of the three cloud providers, load the appropriate cloud-specific version of the Databricks platform to the user's cloud account, and purchase CPU and memory resources directly from the cloud provider. (*Id.* at 18:8-13 (confirming that "when a user configures a cluster in Databricks classic compute, the user is asking to create a cluster using the cloud provider's CPUs and memory within their cloud provided account"); Ex. 4, Davis Op., ¶ 70; Ex. 5, Weissman Rebut., ¶¶ 240, 326-328.) By contrast, in serverless compute, the compute resources are managed by Databricks and the necessary resources run within the user's Databricks account. (Ex. 5, Weissman Rebut., ¶¶ 103, 113-114.)

11. As shown below, the Databricks platform has two data processing engines: Spark and Photon. (*Id.*; Ex. 6, Davis Dep. (Day 1), 83:20-84:2.)



(Ex. 2, Databricks_R2_00091426, 487.)

12. Certain functionality implemented by the Databricks platform's Spark engine comes from open-source Apache Spark. (Ex. 4, Davis Op., ¶ 121; Ex. 5, Weissman Rebut., ¶¶ 85-

[redacted]

87, 93-95.) Spark includes functionality that allows users to run different data processing tasks, including, for example, a "join." (Ex. 4, Davis Op., ¶ 85; Ex. 5, Weissman Rebut., ¶ 165; Ex. 6, Davis Dep. (Day 1), 48:23-49:9.) At a high level, a "join" enables a user to combine data. (Ex. 5, Weissman Rebut., ¶ 166.) When a user chooses to perform a "join," the Databricks platform's Spark engine will use one of five possible strategies for the join: (1) Broadcast Hash Join; (2) Sort Merge Join; (3) Shuffled Hash Join; (4) Cartesian Product Join; and (5) Broadcast Nested Loop Join. (*Id.* at ¶ 180; Ex. 6, Davis Dep. (Day 1), 10:3-11:17.) As explained below, R2's infringement theory requires the use of "Sort Merge Join" specifically. (*Id.* at 10:3-9.) When a user writes code for a query such as "join," the user can also choose the join strategy by providing a "hint" for the specific join strategy in their code. (*Id.* at 78:10-79:9 ("users know how to select a particular join strategy" by "writ[ing] a hint for a shuffle hash join" or any other join strategy).)

13. Photon is a high performance Databricks vectorized query engine in the Databricks platform. (Ex. 5, Weissman Rebut., ¶ 164.) Photon has been the default functionality in the Databricks platform since the launch of Databricks Runtime 9.1 in September 2021. (*Id.*; Ex. 6, Davis Dep. (Day 1), 46:1-3 ("Photon is on by default in the Databricks platform").) One of the key features of Photon includes "replac[ing] sort-merge joins with hash-joins." (Ex. 5, Weissman Rebut., ¶ 164; Ex. 3, GMacartney_00000892); Ex. 6, Davis Dep. (Day 1), 94:12-17 (testifying "[P]hoton [] does not execute the sort-merge join strategy that [Mr. Davis] rel[ied] on for infringement").)

14. "Photon is *enabled by default*[3] on [classic] compute running Databricks Runtime 9.1 LTS and above" but can be disabled by "[de]select[ing] the **Use Photon Acceleration** checkbox." (Ex. 3, GMacartney_00000892, 893; Ex. 6, Davis Dep. (Day 1), 88:14-18 ("for

[3] Unless otherwise noted, all emphasis is added.

[redacted]

Databricks classic compute clusters, photon is on by default"), 47:9-14 ("when [Mr. Davis] created a cluster in the Databricks platform, the [P]hoton acceleration button was checked on, and [he] had to turn it off").) By contrast, "Photon *runs by default* on…serverless compute" and cannot be turned off. (Ex. 3, GMacartney_00000892; Ex. 6, Davis Dep. (Day 1), 92:5-19 (confirming he is "not aware of any functionality…that allows turning photon off for serverless compute").)

C. R2's infringement theory

15. R2 focuses its infringement case on a Spark "Sort Merge Join" within the Databricks platform. (Ex. 6, Davis Dep. (Day 1), 53:25-54:5 ("sort-merge join is the functionality that [he is] accusing."); *see also id.* at 59:12-18, 10:3-9; Ex. 4, Davis Op., ¶¶ 138, 147, 154, 168, 169, 171 (each mapping steps of "Sort Merge Join" to elements of the asserted claim).) Mr. Davis alleges that "Sort Merge Join" was introduced June 1, 2015 as part of Apache Spark 1.4 in the Databricks platform, and that this functionality "is the same across all versions of Apache Spark between version 1.4 and 3.5.1." (Ex. 4, Davis Op., ¶¶ 124, 127, 129.)

16. For his theory, Mr. Davis (1) created a configuration of the Databricks platform for classic compute in which he "chose to turn off photon acceleration" which is "on by default" (Ex. 6, Davis Dep. (Day 1), 45:23-46:3); (2) "created data sets that [he] ran together with [his] Spark 3.5.1 data processing software" (*id.* at 18:15-19; Davis Op., App. G (showing the software he created and relied on for infringement); and (3) ran his created software against his created data sets in his created configuration to force the execution of the "Sort Merge Join" that he mapped to the claims (Ex. 6, Davis Dep. (Day 1), 10:3-9, 53:25-54:5, 59:12-18).

17. But Mr. Davis testified that he does not know whether any Databricks user does what he did. (*Id.* at 123:8-21.) Indeed, he does not "have any Databricks user's data processing code" or "any data sets of any Databricks customers." (*Id.* at 117:13-17, 120:11-13.) And so he "do[es]n't have any direct evidence that use[r]s are executing sort-merge join functionality." (*Id.*

[redacted]

at 125:4-7.) In fact, Mr. Davis conceded that "the Spark sort-merge join functionality that [he] rel[ied] on for infringement will only execute if a Databricks user turns *off* the default photon functionality." (*Id.* at 90:1-7; *see also id.* at 89:16-23.) But he "do[es]n't know whether any Databricks user turns off the default photon functionality in Databricks classic compute." (*Id.* at 116:6-11.) And with the default Photon functionality left on, Mr. Davis's created data processing code "results in executing a [P]hoton shuffled hash join and not the Spark sort-merge join that [he] relied on for infringement."[4] (*Id.* at 89:8-14; Ex. 5, Weissman Rebut., ¶¶ 67, 69-71, 239, 327.) For serverless compute, Mr. Davis was "not aware of any functionality in the Databricks platform that allows turning [P]hoton off." (Ex. 6, Davis Dep. (Day 1), 92:5-9.)

IV. ARGUMENT

A. The Court should grant partial summary judgment because the Databricks platform for serverless compute cannot infringe as a matter of law

R2's expert Mr. Davis alleges without any analysis that "Databricks performs the method outlined in claims 1 and 5 via its 'serverless' offerings" and "Databricks makes the system outlined by claims 17 and 21 by deploying Databricks Runtime to its own cloud environments, such as in connection with Databricks 'serverless' offerings." (Ex. 4, Davis Op., ¶¶ 227-228.) These are the only paragraphs in Mr. Davis's entire report that even allege infringement by serverless compute.[5] Mr. Davis did not map serverless compute to any asserted claim, because he admittedly "d[id]n't know how to interact with serverless compute" and so he "did not test serverless compute to form [his] infringement opinions in this case." (Ex. 6, Davis Dep. (Day 1), 97:7-12, 46:8-11; *see also id.* at 93:2-8 ("I did no testing of serverless compute"), 102:10-14 (same).) Mr. Davis knew only that serverless compute "runs on a Databricks platform" and "d[id]n't actually know whether

[4] Mr. Davis confirmed he "did not provide any infringement opinion in this case that a [P]hoton shuffle hash join strategy meets the…asserted claims." (Ex. 6, Davis Dep. (Day 1), 88:7-12.)
[5] R2 also did not disclose any theory related to serverless compute in its infringement contentions.

[redacted]

serverless compute performs the functionality that [he] rel[ied] on for infringement." (*Id.* at 93:10-15.) This is insufficient as a matter of law. *Fintiv., Inc. v. Apple Inc.*, No. 1:21-cv-896-ADA, 2023 WL 4237356, at *3-5 (W.D. Tex. June 28, 2023) ("mere speculation" by patentee's expert was "insufficient to establish a genuine dispute of material fact that the accused products infringe").

Moreover, the undisputed facts are clear that Mr. Davis has no basis to allege infringement by serverless compute. It is undisputed that serverless compute *runs Photon by default and Photon cannot be turned off*. (Ex. 6, Davis Dep. (Day 1), 102:4-8 ("Databricks serverless compute has [P]hoton enabled by default"), 92:5-15 (Mr. Davis did not "encounter any functionality that allows turning [P]hoton off for serverless compute"); Ex. 3, GMacartney_00000892 ("Photon *runs by default* on…serverless compute"); Ex. 5, Weissman Rebut., ¶ 120.) When Photon is on, which is always the case for serverless compute, it "execut[es] a [P]hoton shuffled hash join and not the Spark sort-merge join that [Mr. Davis] relied on for infringement." (Ex. 6, Davis Dep. (Day 1), 89:8-14; *see also id.* at 94:12-17 ("[P]hoton functionality does not execute the sort-merge join strategy that [Mr. Davis] rel[ied] on for infringement in this case"), 89:16-90:7 (same); Ex. 5, Weissman Rebut., ¶¶ 67, 69-71.) Indeed, one of the key features of Photon includes "replac[ing] sort-merge joins with hash-joins." (Ex. 3, GMacartney_00000892.) And as Mr. Davis conceded, he "did not provide any infringement opinion…that a [P]hoton shuffle hash join strategy meets the requirements of the asserted claims." (Ex. 6, Davis Dep. (Day 1), 88:7-12.)

Mr. Davis's only basis for accusing serverless compute is based on R2's misinterpretation of the source code stipulation in this case. Mr. Davis repeatedly testified "representations [were] made to [him] that open source Apache Spark demonstrates…the relevant functionality of the Databricks Platform" "including Photon." (*Id.* at 104:6-16, 105:7-14.) But the source code stipulation is clearly limited to Spark. Indeed, the parties stipulated that "the open-source Apache Spark source code files identified in R2's First Supplemental Infringement Contentions is

[redacted]

representative of Databricks' corresponding internal Apache Spark source code files in the Databricks Data Intelligence Platform." (Dkt. 73.) Nothing in the stipulation refers to Photon at all, much less states that open-source Spark is representative of the Databricks platform as a whole. R2 cannot rely on a misreading of the source code stipulation to manufacture a fact dispute.[6] Thus, the Court should grant partial summary judgment of no infringement for serverless compute.

B. The Court should grant partial summary judgment because R2 cannot show that any Databricks user directly infringes asserted method claims 1 and 5 under the Federal Circuit's opinion in *Akamai*

Direct infringement of method claims only occurs where "all steps of a claimed method are performed by or attributable to a single entity." *Akamai Techs., Inc. v. Limelight Networks, Inc.*, 797 F.3d 1020, 1022 (Fed. Cir. 2015). There is no evidence that Databricks or any user performs the claimed method in the way Mr. Davis alleges. In fact, Mr. Davis admits that he "do[es]n't have any direct evidence that use[r]s are executing sort-merge join functionality," which he relies on for infringement. (Ex. 6, Davis Dep. (Day 1), 124:20-125:7.) The Databricks customers deposed in this case also testified [redacted]. (Ex. 10, Viswanath (Comcast) Dep., 101:10-16; Ex. 11, Sirois (Abnormal Security) Dep., 79:19-23; *see also* Ex. 6, Davis Dep. (Day 1), 132:9-133:11, 134:18-24.) There is simply no evidence of a direct infringer.

To concoct an infringement theory, Mr. Davis *created* data sets, *created* code, and *created* a specific configuration of the Databricks platform for classic compute. And as Mr. Davis testified, this concocted theory requires no less than a half dozen "conditions under which [he] believe[s] the infringing functionality…can be invoked." (*Id.* at 47:24-48:3.) These conditions are:

- ***First***, the user must "disable [P]hoton to be infringing." (*Id.* at 88:20-89:1.) Mr. Davis

[6] Regardless, interpretation of a court order is a "straightforward legal issue[]" for the Court to determine. *Matter of Burch*, 835 Fed. App'x 741, 747 (5th Cir. 2021).

[redacted]

"chose to turn off [P]hoton acceleration" which is "on by default" (*id.* at 45:23-46:3, 88:14-18) because when Photon is left on, the Databricks platform "execut[es] a [P]hoton shuffled hash join and not the Spark sort-merge join that [Mr. Davis] relied on for infringement" (*id.* at 89:8-14). And Mr. Davis admits he "did not provide any infringement opinion in this case that a [P]hoton shuffle hash join strategy meets the requirements of the asserted claims." (*Id.* at 88:7-12.)

- ***Second***, the user "has to give some code" to "instruct the system" (*id.* at 13:8-14:6) and must run the code on data sets "large enough so that neither [data set] qualifies for a broadcast join," which "doesn't infringe" (*id.* at 67:9-20, 67:25-68:3 (confirming "if one of the data sets is small enough for a broadcast join, there's no infringement")).
- ***Third*****,** the Spark "sort-merge join strategy must be invoked and not one of the other join strategies" as there are at least four alternative join strategies available in Spark. (*Id.* at 58:24-59:18; *see also id.* at 10:3-11:17 (confirming Mr. Davis did not provide any infringement opinion for a Spark Broadcast Hash Join, Shuffle Hash Join, Broadcast Nested Loop Join, or Cartesian Product Join).)
- ***Fourth***, "the join must be performed on data groups that have different schemas" because the asserted claims "require[] two data groups" and further require that the data groups have "different schema." (*Id.* at 61:1-18.) Indeed, neither "a join on one data group" nor "[a] join between two data groups of the same schema" would meet the asserted claims. (*Id.* at 62:8-25; Ex. 1, '610 patent, claims 1, 17.)
- ***Fifth***, the join must be between two data groups that have a key in common. Otherwise, the join "would not meet the asserted claims." (Ex. 6, Davis Dep. (Day 1), 64:4-7.)
- ***Sixth***, "the keys involved in a join [must be] of a data type that supports sorting"

[redacted]

because "to use sort-merge-join, the keys have to be sortable." (*Id.* at 66:1-15.)

If *even one* of Mr. Davis's conditions is not met, there can be no infringement. (*See id.* at 47:24-48:3.) As Mr. Davis admitted, he has no evidence that any Databricks customer or user has met *any* of these conditions, much less *every* condition he applied to manufacture his infringement theory. Mr. Davis testified that (1) he does not "have specific knowledge" of whether "any Databricks user turns off the default [P]hoton functionality in Databricks classic compute" (*id.* at 116:1-11); (2) he is "not aware of any evidence in this case about the size of the data sets that Databricks users implement joins on," and thus, does not know whether they would qualify for a non-infringing Broadcast Hash Join (*id.* at 121:4-9); (3) he does not know "what type of join strategies any Databricks user implements," including whether any user implements the Sort Merge Join strategy he accuses as opposed to any of the other unaccused join strategies (*id.* at 117:25-118:3); (4) he does not know "whether [any Databricks customers or users] implement a join onto data groups with the same schema" or different schemas (*id.* at 119:6-11, 119:21-120:1); (5) he does not know "for any Databricks customers or users, whether they implement a join between two data groups that share a key in common" (*id.* at 119:13-120:1); and (6) he does not know "whether Databricks users or customers implement data sets that have keys that are of a type that supports sorting" (*id.* at 122:8-14). As Mr. Davis further testified, he has no evidence of any "Databricks users' data processing code in this case" (*id.* at 117:13-17) and "do[es] not have any data sets of any Databricks customers in this case" (*id.* at 120:11-13). Mr. Davis simply does not know "what customers do or don't do." (*Id.* at 117:19-23.)

Additionally, an entity is only responsible for another's performance of claimed method steps "(1) where that entity directs or controls others' performance, and (2) where the actors form a joint enterprise." *Akamai*, 797 F.3d at 1022. But Mr. Davis provides no evidence that Databricks "directs or controls" any user to "do what [he] did to show infringement." (Ex. 6, Davis Dep. (Day

[redacted]

1), 123:8-12.) It is an undisputed fact that each Databricks user chooses how to implement the Databricks platform. (Ex. 10, Viswanath Dep., 92:3-10 ([redacted]); Ex. 11, Sirois Dep., 72:22-73:2 [redacted]), 73:19-74:2 ([redacted]); Ex. 5, Weissman Rebut., ¶ 91.) Mr. Davis also fails to identify any evidence of a "joint enterprise," which does not exist.

At best, Mr. Davis alone is a direct infringer under his concocted theory, which is insufficient as a matter of law. *ACCO Brands, Inc. v. ABA Locks Mfrs. Co.*, 501 F.3d 1307, 1313 (Fed. Cir. 2007) (finding "no evidence of direct infringement" where plaintiff's expert was the only witness to testify about performing the infringing steps and "the record contains no evidence of actual users having operated [the accused product] in an infringing manner"). Thus, the Court should grant partial summary judgment of no direct infringement of method claims 1 and 5.

C. The Court should grant partial summary judgment because Databricks classic compute does not directly infringe asserted system claims 17 and 21 as a matter of law under the Federal Circuit's opinion in *Centillion*

Databricks can only directly infringe system claims 17 and 21 if it "makes," "uses," or "sells" the claimed invention. *Centillion Data Sys., LLC v. Qwest Commc'ns Int'l, Inc.*, 631 F.3d 1279, 1288 (Fed. Cir. 2011). Here, the Court construed the preamble of claim 17 as limiting, which requires, in part, a "computer system configured to process data of a data set, wherein…the computer system comprises at least one processor and memory that are operable to perform" the MapReduce data processing steps of the claim. (Dkt. 71 at 6; Ex. 1, '610 patent, claim 17.) To "make" or "sell" the claimed system, Databricks must "combine all of the claim elements" including the "processor and memory." *Centillion*, 631 F.3d at 1288. To "use" the claimed systems, Databricks must "put the invention into service, *i.e.*, control the system as a whole and

[redacted]

obtain benefit from it." *Id.* at 1284. R2 has no evidence that Databricks does either.

Databricks does not "make" the claimed "computer system" with the required "processor and memory that are operable to perform" the MapReduce steps of the claim. R2 does not even allege, much less provide any evidence that Databricks "combines all of the elements" of asserted claims 17 and 21. Mr. Davis only states that "[t]he Databricks platform *must* include at least one processor and memory." (Ex. 4, Davis Op., ¶ 188.) But as Mr. Davis conceded, "Databricks itself does not provide processors or memory resources for classic compute." (Ex. 6, Davis Dep. (Day 1), 17:22-18:7; *see also* Ex. 5, Weissman Rebut., ¶ 109.) The user must create an account with a third-party cloud provider such as Google Cloud or Amazon AWS, load the Databricks platform into that cloud account, and "create a cluster using the cloud provider's CPUs and memory" in that account. (Ex. 6, Davis Dep. (Day 1), 18:10-11, 7:22-8:1.) But even then, the claimed "computer system" has not yet been "configured to process data of a dataset" nor has the cloud provider's "processor and memory" been made "operable to perform" the MapReduce steps of the claim. (Ex. 1, '610 patent, claim 17.) The user must also provide data sets and data processing code, and then "click run now" in the Databricks platform, which only at that point causes the user's provided "data processing code…to run on the AWS cloud resources, processors, and memory." (Ex. 6, Davis Dep. (Day 1), 16:13-19; *see also id.* at 13:8-14:6 (confirming user must provide data processing code), 151:9-15 ("The user has to define the data.").) Thus, to the extent the claimed "computer system" is ever "made," only a Databricks user combines all the claim elements—*not* Databricks—which is insufficient as a matter of law. *Centillion*, 631 F.3d at 1288. For the same reasons, Databricks does not "sell" the claimed system. Databricks does not sell the claimed "computer system," "processor and memory," or the required data sets and data processing code that a user must provide to make the "computer system" operable to perform the claimed MapReduce steps. Moreover, for classic compute, users "pay the cloud provider directly for their

[redacted]

usage of the cloud provider's resources." (Ex. 6, Davis Dep. (Day 1), 17:22-18:2.) Because Databricks "may not be held liable…for 'making' or 'selling' less than a complete invention," Databricks cannot directly infringe claims 17 and 21 as a matter of law. *Rotec Indus., Inc. v. Mitsubishi Corp.*, 215 F.3d 1246, 1252 n.2 (Fed. Cir. 2000).

Databricks also does not "use" the claimed system as it does not "put the invention into service, *i.e.*, control the system as a whole and obtain benefit from it." *Centillion*, 631 F.3d at 1284. To show that Databricks "controls" the claimed system, R2 needs to show Databricks "use[s] each and every element of [the] claimed system." *Id.* R2 cannot. Indeed, as discussed in Section IV.B., Mr. Davis conceded he "do[es]n't have any direct evidence that" "any Databricks user does what [he] did to show infringement." (Ex. 6, Davis Dep. (Day 1), 123:8-12.) Moreover, Databricks does not "[redacted]" (Ex. 7, Cody Davis Dep., 163:11-14; Ex. 10, Viswanath Dep., 92:3-10 ([redacted]); Ex. 11, Sirois Dep., 72:22-73:2 [redacted]), 73:19-74:2 ([redacted]).) Databricks users choose their own data processing configuration by providing their own data sets, creating their own data processing code, and configuring the cloud provider's processors and memory. (Ex. 6, Davis Dep. (Day 1), 13:8-14:6, 16:13-19, 151:9-15.) R2 does not contend otherwise because it cannot. As Mr. Davis conceded, he "do[es]n't have specific knowledge of customers' configurations," "ha[s]n't reviewed Databricks customers queries and their source code," and does not know "what customers do or don't do." (*Id.* at 116:1-5; 117:4-23.) There is no evidence Databricks or any user directly infringes through "use" of the claimed system.

[redacted]

D. R2 cannot show the accused functionality meets the requirements of the claimed "data groups" as construed by the Court

Every asserted claim requires a "plurality of data groups" where "the data of a first data group has a different schema than the data of a second data group." (Ex. 1, '610 patent, claims 1, 17.) The Court construed "data group" as "a group of data *and* a mechanism for identifying data in that group." (Dkt. 71 at 6.) Thus, to meet the Court's construction, each recited "data group" must include *both* "a group of data *and*" the required "mechanism…." R2's expert Mr. Davis accuses Delta tables as the claimed "data groups." But Mr. Davis identifies nothing in a Delta table that meets the requirement of "a mechanism for identifying data in that group." (Ex. 4, Davis Op., ¶ 137.) This alone is fatal to R2's infringement case.

Mr. Davis asserts the "'mechanism for identifying data from that group' is the schema of the table itself" and that "schemas [are] represented by Attribute IDs." (*Id.* at ¶¶ 137, 139; *see also* Ex. 6, Davis Dep. (Day 1), 209:15-17 ("schema is the mechanism for identifying data from a data group"), 208:12-14 ("schema consists of multiple Attribute IDs").) In other words, it is Mr. Davis's opinion that a "Delta table" is the claimed "data group" and Attribute IDs are the "data group's" "mechanism for identifying data in that group." But Attribute IDs have nothing to do with a Delta table, and thus, cannot be the "mechanism for identifying data from a [Delta table]."

In fact, a Delta table is simply a way for Databricks users to "store data in tables" in the Databricks platform. (Ex. 5, Weissman Rebut., ¶127; Ex. 4, Davis Op., ¶ 91 ("data is loaded into Delta tables").) Neither DataFrames nor Attribute IDs exist when a Delta table is created. When Spark is executing a query, it "draws data from various sources [such as a Delta table] into…DataFrames"[7] (Ex. 4, Davis Op., ¶ 75) and creates "Attribute IDs [that] refer to the columns

[7] In Spark, a DataFrame "is a Dataset organized into named columns," which is created when a user executes a query. (Ex. 4, Davis Op., ¶ 75; Ex. 6, Davis Dep. (Day 1), 165:2-24.)

[redacted]

of a particular DataFrame" (Ex. 6, Davis Dep. (Day 1), 137:21-23). This is accomplished during the first step of executing a Spark query, which "reads [] data sets…from [D]elta tables" to "create[] the corresponding DataFrames" as well as "the Attribute IDs." (*Id.* at 165:2-24.) Indeed, it is undisputed that both a DataFrame and its "Attribute IDs are created…during query execution in the Databricks platform." (*Id.* at 135:6-10; *see also id.* at 165:2-24.)

But Mr. Davis admitted, "*data groups exist prior to* the formation of []Dataframes" (Ex. 4, Davis Op., ¶ 205), and thus, "a DataFrame is not the claimed data group" (Ex. 6, Davis Dep. (Day 1), 215:20-23). Because a DataFrame and its Attribute IDs are created *during* query execution, and do not exist as part of a Delta table (the alleged "data group"), neither can meet the requirements of a "data group" including the required "mechanism…." (*Id.* at 164:21-165:25.) The Court should grant summary judgment because there is no dispute of fact the accused "data groups" (Delta tables) do not include the required "mechanism for identifying data in that group."

E. R2 cannot show the accused functionality meets the requirements of the specialized "mapping functions" recited by the claims

During IPR proceedings, R2 distinguished the prior art, in part, by arguing that "the '610 patent claims…*require[] the mapping functions to be specialized* so that they can be 'selected' for a partition based on the 'data group' the partition originated from…." (Dkt. 71 at 24.) The Court found this statement "r[o]se to the level of a 'clear and unmistakable' disclaimer of claim scope," and thus construed the "providing…" limitation of the claim to mean "providing each data partition to one of a plurality of different mapping functions where each mapping function is selected for a partition based on the data group the partition originated from." (*Id.*)

As explained in Section IV.D., Mr. Davis accuses Delta tables as meeting the claimed "data groups." (Ex. 4, Davis Op., ¶¶ 137-138.) Additionally, Mr. Davis accuses the Spark operator ShuffleExchangeExec as the claimed "mapping function." (Ex. 6, Davis Dep. (Day 1), 128:11-14

[redacted]

("each instantiation of ShuffleExchangeExec is one of the claimed mapping functions"); Ex. 4, Davis Op., ¶ 151 (identifying "ShuffleExchangeExec instantiations (mapping functions)").) Thus, to prove infringement under the Court's construction, Mr. Davis must show that "each [ShuffleExchangeExec instantiation] is selected for a partition based on the [Delta table] the partition originated from." (*See* Dkt. 71 at 24.)

But Mr. Davis admitted this is not how the Databricks platform works. Mr. Davis testified that ShuffleExchangeExec "does not check whether a partition comes from a particular delta table." (Ex. 6, Davis Dep. (Day 1), 173:15-19.) And he did not stop there. Mr. Davis further conceded that ShuffleExchangeExec "does not consider or use [D]elta tables" and "has no relationship with Delta Lake tables." (*Id.* at 173:2-13.) Databricks staff software engineer Josh Rosen confirmed this. Mr. Rosen testified that t[redacted] (Ex. 8, Rosen Dep., 194:19-23; *see also* Ex. 5, Weissman Rebut., ¶¶ 52, 257.) There is no dispute of fact that the Court's construction is not met, and thus, the Court should grant summary judgment.

F. R2 cannot show the accused functionality meets the requirements of the claimed "reducing"

Every asserted claim requires that the claimed "reducing…includ[es] processing the intermediate data for each data group in a manner that is defined to correspond to *that data group*." (Ex. 1, '610 patent, claims 1, 17.) During IPR proceedings, R2 interpreted "that data group" as referring to "the data group from which the intermediate data originated." (Dkt. 71 at 29-30.) Based on this admission, the Court construed the "[processing] / [process]…" limitation of the claim to mean "[processing] / [process] the intermediate data for each data group in a manner that is defined to correspond to the data group from which the intermediate data originated." (*Id.*)

As explained above in Section IV.D., Mr. Davis accuses Delta tables of meeting the

[redacted]

claimed "data groups." (Ex. 4, Davis Op., ¶¶ 137-138.) And Mr. Davis accuses the Spark SortMergeJoinExec operator as the claimed "reducing." (Ex. 4, Davis Op., ¶ 142; Ex. 6, Davis Dep. (Day 1), 129:20-25.) Thus, to prove infringement, Mr. Davis must show that SortMergeJoinExec (alleged "reducing") "include[es] processing the intermediate data for each [Delta table] in a manner that is defined to correspond to that [Delta table]." (*See* Dkt. 71 at 30.)

But Mr. Davis again admitted that this is not how the Databricks platform works. Mr. Davis testified that "none of the processing in the sort-merge join step considers or uses [D]elta tables." (Ex. 6, Davis Dep. (Day 1), 178:7-10; *see also id.* at 178:12-17 (explaining SortMergeJoinExec "does not consider whether the outputs of the sort step [(the "intermediate data")] come from a particular [D]elta table"), 178:18-23 (explaining SortMergeJoinExec "does not consider whether the left and right inputs come from a particular [D]elta table").) In other words, there is no dispute that the accused "reducing" does not consider the accused "data group" during processing, nor is the "reducing" *defined* to correspond to the "data group" from which intermediate data originated. (Ex. 5, Weissman Rebut., ¶¶ 53, 258.) In fact, Mr. Rosen testified that [redacted] (Ex. 8, Rosen Dep., 195:12-16.) And in the context of discussing the accused SortMergeJoinExec functionality, Mr. Xin (Databricks co-founder) testified [redacted] (Ex. 9, Xin Dep., 129:21-130:7.) In fact, [redacted] (*Id.* at 318:4-9, 317:2-16.) Thus, the Court should grant summary judgment as there is no dispute of fact that the Court's construction is not met.

V. CONCLUSION

For the foregoing reasons, the Court should grant summary judgment of non-infringement.

[redacted]

Dated: April 10, 2025

Respectfully submitted,

/s/ Vigen Salmastlian

Michael J. Sacksteder
CA Bar No. 191605 (Admitted E.D. Texas)
Email: msacksteder@fenwick.com
Gregory Sefian
CA Bar No. 341802 (Admitted *Pro Hac Vice*)
Email: gsefian@fenwick.com
S. Emma Lee
CA Bar No. 344074 (Admitted *Pro Hac Vice*)
Email: emma.lee@fenwick.com
FENWICK & WEST LLP
555 California Street, 12th Floor
San Francisco, CA 94104
Telephone: 415.875.2300
Facsimile: 415.281.1350

Vigen Salmastlian
CA Bar No. 276846 (Admitted E.D. Texas)
Email: vsalmastlian@fenwick.com
FENWICK & WEST LLP
801 California Street,
Mountain View, CA 94041
Telephone: 650.988.8500
Facsimile: 650.938.5200

Jessica M. Kaempf
WA Bar No. 51666 (Admitted E.D. Texas)
Email: jkaempf@fenwick.com
Jonathan G. Tamimi
WA Bar No. 54858 (Admitted E.D. Texas)
Email: jtamimi@fenwick.com
FENWICK & WEST LLP
401 Union Street, 5th Floor
Seattle, WA 98101
Telephone: 206.389.4510
Facsimile: 206.389.4511

Dargaye Churnet
CA Bar No. 303659 (Admitted E.D. Texas)
Email: dchurnet@fenwick.com
FENWICK & WEST LLP
730 Arizona Ave, 1st Floor
Santa Monica, CA 90401

[redacted]

Telephone: 310.434.5400

Jennifer Truelove
Texas Bar No. 24012906
Email: jtruelove@mckoolsmith.com
MCKOOL SMITH P.C.
104 E. Houston Street, Suite 300
Marshall, TX 75670
Telephone: 903.923.9000
Facsimile: 903.923.9099

Derron R. Dacus
Texas Bar No. 00790553
Email: ddacus@dacusfirm.com
THE DACUS FIRM, P.C.
821 ESE Loop 32, Suite 430
Tyler, Texas 75701
Telephone: 903.705.1117

Attorneys for Defendant
Databricks, Inc.

[redacted]

CERTIFICATE OF AUTHORIZATION TO SEAL

Pursuant to Local Rule CV-5(a)(7)(B) that I am authorized to file the foregoing document under seal pursuant to Protective Order (Dkt. 61) because this document references Designated Material.

/s/ Vigen Salmastlian
Vigen Salmastlian

CERTIFICATE OF SERVICE

The undersigned hereby certifies that on April 10, 2025, a true and correct copy of the above and foregoing document has been served on all counsel of record who are deemed to have consented to electronic service via the Court's CM/ECF system. Additionally, I hereby certify that all counsel of record who have consented to electronic service are being served with a copy of these documents via electronic mail per Local Rule CV-5.

/s/ Vigen Salmastlian
Vigen Salmastlian